In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from PIL import Image
import import_ipynb
import io
import fidelity

im = Image.open('house.tif')
f = np.array(im)
```

In [6]:
```
dim1, dim2 = np.shape(f)
gamma = 2.2
fl = 255*((f/255)**gamma)
```

In [7]:
```
T = 127
out = np.zeros((dim1,dim2))
out = np.pad(out,((1,1),(1,1)))
fl = np.pad(fl,((1,1),(1,1)))

for i in range(1,dim1+1):
    for j in range(1,dim2+1):
        # apply quantization
        if fl[i,j] > T:
            out[i,j] = 255
        else:
            out[i,j] = 0
        # compute quantization error
        pixelError = fl[i,j] - out[i,j]
        # diffusing error to other indices
        fl[i+1,j-1] += pixelError*(3/16)
        fl[i+1,j] += pixelError*(5/16)
        fl[i+1,j+1] += pixelError*(1/16)
        fl[i,j+1] += pixelError*(7/16)
# remove padding from output matrix
out = out[1:-1,1:-1]

# display image
# plt.figure()
# plt.title("Error Diffusion Halftone of house.tif")
# plt.imshow(out,cmap=plt.cm.gray,interpolation='none')

# save image as .tif
img_out = Image.fromarray(out.astype(np.uint8))
img_out.save("ErrorDiffusion.tif")
```

In [8]:
```
fid = fidelity.fidelity(f, out)

# compute RMSE
count = 0
for i in range(dim1):
    for j in range(dim2):
        count += (f[i,j] - out[i,j])**2
RMSE = np.sqrt((1/(dim1*dim2))*count)
```

```
# display RMSE and fidelity
print("RMSE = ", RMSE)
print("fidelity = ", fid)
```

```
RMSE =  98.84711671109255
fidelity =  13.427253039026654
```